saschavanessen@gmail.com

Python Course

Week 2:
Make choices and reuse code

Workshop Overview

1. Writing your first program
2. Making choices and reuse code
3. Loops and strings
4. Files and lists
5. Dictionaries and tuples

Acknowledgments: Structure of the workshop follows the book “Python for informatics” by Charles Severance. Several examples are from this book or the accompanying slides.

Python Background

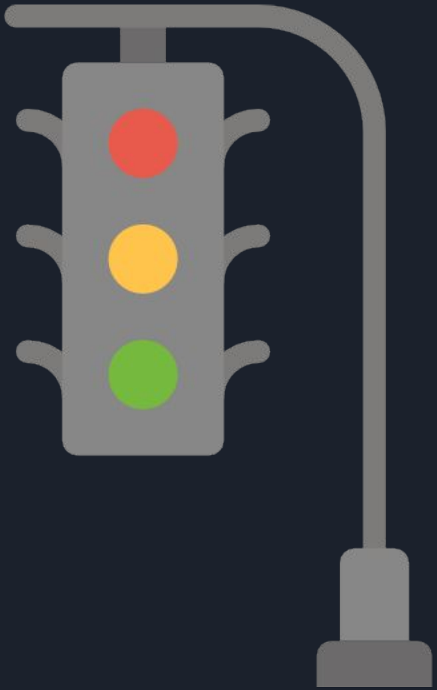
- Guido van Rossum
- “Hobby” project around Christmas
- While working at CWI
- “Python aims to encourage the creation of reusable code”
- Name origin: Monty Python https://youtu.be/sXE8LdXzeHM

https://www.python.org/doc/essays/foreword/

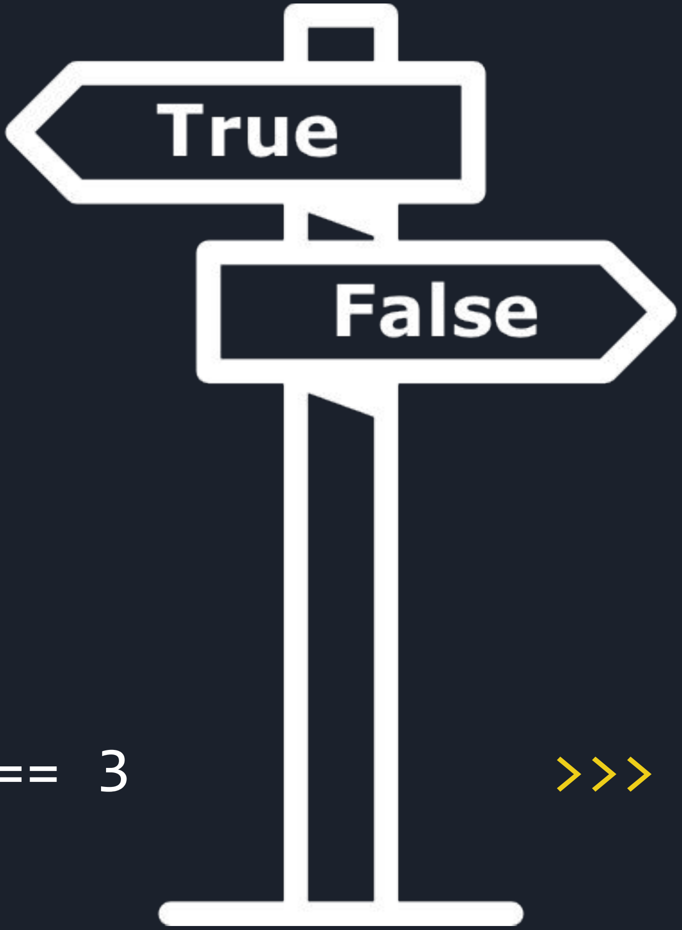
Making Choices

Python and indentation

```
if traffic_light == ‘green’:
  print(“Go! Go! Go!”)
else:
  print(“Stop!”)
```



Datatype: boolean



```
>>> 1 + 2 == 3
```

```
>>> 1 + 1 == 3
```

Comparison operators

```
x == y       # x is equal to y
x != y       # x is not equal to y
x > y        # x is greater than y
x < y        # x is less than y
x >= y       # x is greater than or equal to y
x <= y       # x is less than or equal to y
x is y       # x is the same as y
x is not y   # x is not the same as y
```

Equal or the same?

Equal:

```
>>> 1.0 == 1
True
```

Same:

```
>>> 1.0 is 1
False
```

Logical operations

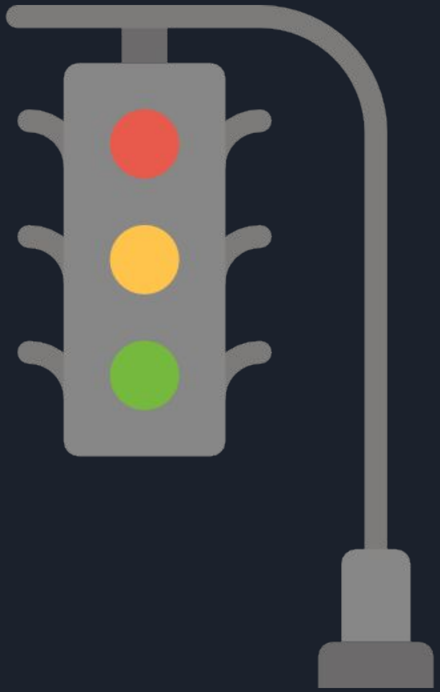
```
x > 0 and x < 10     # both need to be true

x == 0 or y == 0     # one needs to be true

not (x > y)          # negates the expression
                     (“flips” the answer)
```

Python and indentation

```
if traffic_light == ‘green’:
   print(“Go! Go! Go!”)
else:
   print(“Stop!”)
```



not necessarily negative!

```
if x > 0:
   print(“x is positive”)
else:
   print(“x is not positive”)
```

What will happen here?

This is False

Change it to x >= 2

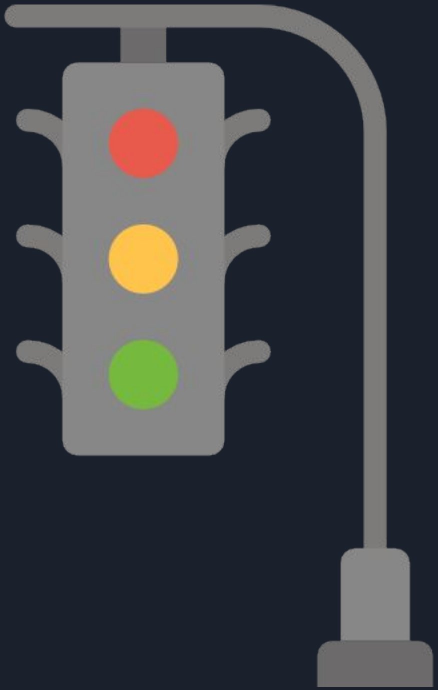
```
x = 2
if x > 2:
   print(“This is in the if-block”)
   print(“and this too”)
print(“Finished”)
```

Both lines in the intended block are executed when the if-statement is True or skipped when the if-statement is False

This is not part of the if-block and in this case always executed

Making choices: if elif else

```
if traffic_light == ‘green’:
    print(“Go! Go! Go!”)
elif traffic_light == “orange”:
    print(“Stop if possible”)
else:
    print(“Stop!”)
```



Think about the order!

```
x = 25
if x < 10:
   print(“Number is below ten”)
elif x < 20:
   print(“Number is below twenty”)
elif x < 40:
   print(“Number is below forty”)
elif x < 30:
   print(“Number is below thirty”)
else:
   print(“Number is forty or higher”)
```

Nested conditionals

```
answer = input(“Do you want something to ‘drink’ or ‘eat’?”

if answer == “drink”:
    drink = input(“What drink would you like?”)
    if drink == “coffee”
        print(“There you go. Coffee for you”)
    elif drink == “tea”:
        print(“Tea for you it is”)
    else:
        print(“Sorry, we don’t have”, drink)
elif answer == “eat”:
    print(“I’ll get you something to eat!”)
else:
    print(“I guess you’re not hungry or thirsty”)
```

Catch exceptions

```
inp = input('Enter Fahrenheit Temperature:')
fahr = float(inp)
cel = (fahr - 32.0) * 5.0 / 9.0
print(cel)
```

Run this program, and enter: 72
Run it again and now enter: blah

Catch exceptions

```
inp = input(“Enter Fahrenheit Temperature:”)
try:

   fahr = float(inp)
   cel = (fahr - 32.0) * 5.0 / 9.0
   print(cel)

except:
    print(“Please enter a number”)
```

Run this program, and enter: 72
Run it again and now enter: blah

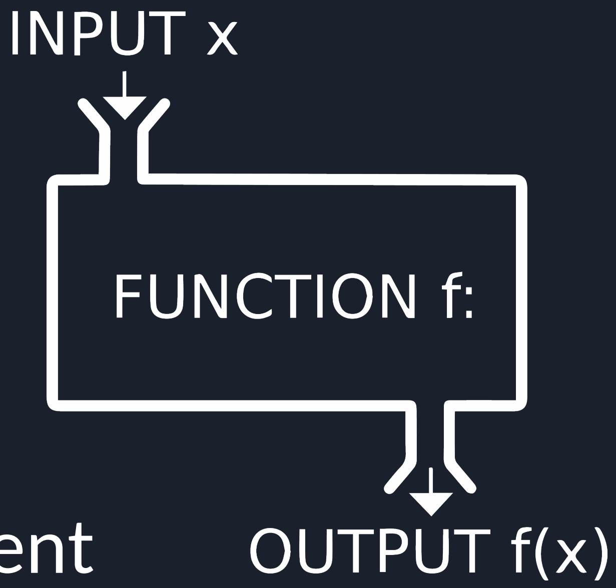
Reuse Code

Functions

- input("What is the answer to everything?")
- type(42)
- int(2.15)

Function name

Arguments / Input



A function "takes" an argument and "returns" a result

Built-in functions

- input()
- str(13)   int(‘13’)   float(13)
- Many more!

```
>>> len(“qwertyuiop”)
>>> min(“qwertyuiop”)
>>> max(“qwertyuiop”)
```

Libraries / modules

- You can import a library
- Each library has functions with a certain theme

```
import random

x = random.random()
y = random.random()
z = random.random()
print(x, y, z)
```

The “random” library

```
>>> import random
```

```
>>> random.randint(1, 6)
```

```
>>> t = [“a”, “b”, “c”]
>>> random.choice(t)
```

The “math” library

```
>>> import math


>>> math.sqrt(9)

>>> math.log(8,2)

>>> math.pi
```

Create your own function

```
def addTwo (a,b):
    added = a + b
    return added

x = addTwo(3,5)
print(x)
```

Why functions?

- Easier to read and understand
- Smaller, eliminate repetitive code
- Easier to debug
- You can reuse it, also in other programs

Calling functions from functions

```
def print_couplet ():
    print("Ik heb een potje met vet,")
    print("Al op de tafel gezet.")
    print("Ik heb een potje potje potje potje ve-e-et,")
    print("Al op de tafel gezet.\n")

def sing_song ():
    print("Dit is het eerste couplet.")
    print_couplet()
    print("Dit is het tweede couplet.")
    print_couplet()

sing_song()
```

In- and output

```
def water_needed (flour):
    water = flour * 325 / 500.0
    return(water)
```

Standard bread recipe:

- 500g flour
- 15g fresh yeast
- 10g sugar
- 10g salt
- 325ml water

```
x = 1000
print(“You need”, water_needed(x), “ml water”)
```

How to install a Python library

Where can I find documentation / help?

- https://docs.python.org/
- https://stackoverflow.com/
- Just search for it! ;)
  - “Python library random”

How to install Python libraries

- https://stackoverflow.com/questions/4750806/how-do-i-install-pip-on-windows
- https://stackoverflow.com/questions/21222114/how-do-i-install-python-libraries

Assignment! Lootjes trekken 1

- Make a list with names of your friends (at least 10 names)
- Choose a name from this list at random
- Print the name

Hints:

- Use the “random” library
- Look at earlier slides

Assignment! Lootjes trekken 2

- Make a function that:
    - Gets as input a list
    - Chooses a name at random from that list
    - Prints the name
    - Removes the name from the list
- Execute the function 3 times

Hints:

- Look up how you can remove an item from a list: “python remove item from list”

Assignment! Baking bread

- You have a certain amount of flour, yeast, sugar, salt and water
- Calculate how many breads you can make

Hints:

- Create a function
- Do the calculations
- Use if-statements
    - Or more advanced: try the min() function

Standard bread recipe:

- 500g flour
- 15g fresh yeast
- 10g sugar
- 10g salt
- 325ml water

To be continued!

- More practice:
  - Exercises in chapter 3 and 4 of the book

- Next week: Loops and strings
- See you next week!

Icons Acknowledgements

- https://www.flaticon.com/authors/smashicons
- https://www.flaticon.com/authors/dimi-kazak
- http://www.freepik.com

Shortcuts

- Terminal

Up	previous commands
Tab	autocomplete

- Editor

Ctrl-/ or Cmd-/	comment on/off
Tab	indent forward
Shift-Tab	indent backwards